CLAIMS

What is claimed is:

1. A system operable to perform a multi-programming-language compilation process on a computer program, comprising:

a compiler framework operable to perform a programming language-independent portion of the compilation process;

a plurality of language modules, wherein each language module is operable to perform a programming language-dependent portion of the compilation process; and

a plurality of language interfaces, wherein each language interface is provided by a language module to interact with the compiler framework.

2. The system of claim 1, wherein:

a multi-threading service is used by the compiler framework and the plurality of language modules.

3. The system of claim 1, wherein:

the system is tailored for the Java environment.

4. The system of claim 1, wherein:

the computer program is organized into a project, which may contain at least one set of the following: files, paths, libraries, configuration information, and dependencies among files.

5. The system of claim 4, further comprising a type cache operable to:

store types defined in the set of files in the project;

store dependencies between the types it stores; and

allow types defined in one programming language to reference types defined in another programming language.

6. The system of claim 4, wherein:

the programming language-independent portion of the compilation process comprises at least one of the following phases:

managing the set of files in the project;

persisting the set of paths files in the project;

maintaining the set of dependencies in the project;

acquiring configuration information files in the project; and

maintaining a list of errors related to the project.

7. The system of claim 1, wherein:

the programming language-dependent portion of the compilation process comprises at least one of the following phases:

performing lexical analysis;

performing syntactic analysis;

performing name resolution;

performing semantic analysis; and

performing code generation.

8. The system of claim 7, wherein:

a language interface of the plurality of language interfaces may present language-dependent portion of the compilation process in the form of a set of components, each component performing one phase of the compilation process.

9. The system of claim 1, wherein:

a language interface of the plurality of language interfaces may include functions for retrieving information about a particular file in the computer program.

10. The system of claim 1, wherein:

a language interface of the plurality of language interfaces allows a first language module of the plurality of language modules to interact with a second language module of the plurality of language modules.

11. The system of claim 10, wherein:

the language interface is operable to nested languages, allowing the first language module to request the compilation of a specified portion of the computer program using the second language module.

12. The system of claim 1, wherein:

at least one language module of the plurality of language modules is for Java language.

13. The system of claim 1, wherein:

a second language module of the plurality of language modules extends a first language module of the plurality of language modules to provide a new language which is an extended version of a programming language compiled by the first language module.

14. The system of claim 1, further comprising:

a tool to speed the development of the plurality of language modules.

15. A system operable to perform a multi-programming-language compilation process on a computer program, comprising:

a compiler framework operable to perform a language-independent portion of the compilation process;

a plurality of language modules, each language module in the plurality of language modules is operable to perform a language-dependent portion of a compilation process and to provide language information about the computer program;

an information interface operable to permit each client in a plurality of clients to interact with the compiler framework;

a plurality of clients operable to utilize the information interface to request the compiler framework to perform at least one of the following tasks:

obtaining the language information produced by the plurality of language modules; and

requesting a service provided by the compiler framework.

16. The system of claim 15, wherein the plurality of clients comprise:

an integrated development environment (IDE) for developing a computer program wherein the IDE uses the language information from the compiler framework to provide language features based on that information; and

a command-line interface for causing the compilation of the computer program and displaying any errors in the computer program.

17. The system of claim 16, wherein:

the IDE may include a source code editor to edit files in the computer program.

18. The system of claim 15, wherein:

the information interface is further operable to allow a client to inform the compiler network of file changes; and

the compiler network is operable to recompile the changed files and other files depend on them.

19. A method operable to perform a multi-programming-language compilation process on a computer program, comprising:

utilizing a compiler framework operable to perform a programming language-independent portion of the compilation process;

invoking a plurality of language modules, wherein each language module is operable to perform a programming language-dependent portion of the compilation process; and

providing a plurality of language interfaces, wherein each language interface is operable to allow a language module to interact with the compiler framework.

20. The method of claim 19, further comprising:

utilizing a multi-threading service during the compilation process.

21. The method of claim 19, further comprising:

tailoring the compilation process for the Java environment.

22. The method of claim 19, wherein:

the computer program is organized into a project, which may contain at least one set of the following: files, paths, libraries, configuration information, and dependencies among files.

23. The method of claim 22, further comprising:

utilizing a type cache operable to:

store types defined in the set of files in the project;

store dependencies between the types it stores; and

allow types defined in one programming language to reference types defined in another programming language.

24. The method of claim 22, wherein:

the programming language-independent portion of the compilation process comprises at least one of the following phases:

managing the set of files in the project;

persisting the set of paths files in the project;

maintaining the set of dependencies in the project;

acquiring configuration information files in the project; and

maintaining a list of errors related to the project.

25. The method of claim 19, wherein:

the programming language-dependent portion of the compilation process comprises at least one of the following phases:

performing lexical analysis;

performing syntactic analysis;

performing name resolution;

performing semantic analysis; and

performing code generation.

26. The method of claim 25, further comprising:

presenting the language-dependent portion of the compilation process in the form of a set of components, each component performing one phase of the compilation process.

27. The method of claim 19, further comprising:

retrieving information about a particular file in the computer program via a language interface of the plurality of language interfaces.

28. The method of claim 19, further comprising:

allowing a first language module of the plurality of language modules to interact with a second language module of the plurality of language modules.

29. The method of claim 28, wherein:

the first language module is operable to request the compilation of a specified portion of the computer program using the second language module.

30. The method of claim 19, wherein:

at least one language module of the plurality of language modules is for Java language.

31. The method of claim 19, further comprising:

extending a first language module of the plurality of language modules using a second language module of the plurality of language modules to provide a new language which is an extended version of a programming language compiled by the first language module.

32. The method of claim 19, further comprising:

adopting tools to speed up the development of the plurality of language modules.

33. A method operable to perform a multi-programming-language compilation process on a computer program, comprising:

utilizing a compiler framework operable to perform a language-independent portion of the compilation process;

invoking a plurality of language modules, each language module in the plurality of language modules is operable to perform a language-dependent portion of a compilation process and to provide language information about the computer program;

providing an information interface operable to permit each client in a plurality of clients to interact with the compiler framework;

including a plurality of clients operable to utilize the information interface to request the compiler framework to perform at least one of the following tasks:

obtaining the language information produced by the plurality of language modules; and

requesting a service provided by the compiler framework.

34. The method of claim 33, wherein the plurality of clients comprise:

an integrated development environment (IDE) for developing a computer program wherein the IDE uses the language information from the compiler framework to provide language features based on that information; and

a command-line interface for causing the compilation of the computer program and displaying any errors in the computer program.

35. The method of claim 34, further comprising:

including a source code editor in the IDE to edit files in the computer program.

36. The method of claim 33, further comprising:

allowing a client to inform the compiler network of file changes; and

recompiling the changed files and other files depend on them.

37. A machine readable medium having instructions stored thereon that when executed by a processor cause a system to:

perform a multi-programming-language compilation process on a computer program, comprising:

utilizing a compiler framework operable to perform a programming language-independent portion of the compilation process;

invoking a plurality of language modules, wherein each language module is operable to perform a programming language-dependent portion of the compilation process; and

providing a plurality of language interfaces, wherein each language interface is operable to allow a language module to interact with the compiler framework.

38. The machine readable medium of claim 37, further comprising instructions that when executed cause the system to:

use a multi-threading service during the compilation process.

39. The machine readable medium of claim 37, further comprising instructions that when executed cause the system to:

tailor the compilation process for the Java environment.

40. The machine readable medium of claim 37, wherein:

the computer program is organized into a project, which may contain at least one set of the following: files, paths, libraries, configuration information, and dependencies among files.

41. The machine readable medium of claim 40, further comprising instructions that when executed cause the system to:

utilize a type cache operable to:

store types defined in the set of files in the project;

store dependencies between the types it stores; and

allow types defined in one programming language to reference types defined in another programming language.

42. The machine readable medium of claim 40, further comprising instructions that when executed cause the system to:

perform the programming language-independent portion of the compilation process in at least one of the following phases:

managing the set of files in the project;

persisting the set of paths files in the project;

maintaining the set of dependencies in the project;

acquiring configuration information files in the project; and

maintaining a list of errors related to the project.

43. The machine readable medium of claim 37, further comprising instructions that when executed cause the system to:

perform the programming language-dependent portion of the compilation process in at least one of the following phases:

performing lexical analysis;

performing syntactic analysis;

performing name resolution;

performing semantic analysis; and

performing code generation.

44. The machine readable medium of claim 43, further comprising instructions that when executed cause the system to:

present the language-dependent portion of the compilation process in the form of a set of components, each component performing one phase of the compilation process.

45. The machine readable medium of claim 37, further comprising instructions that when executed cause the system to:

retrieve information about a particular file in the computer program via a language interface of the plurality of language interfaces.

46. The machine readable medium of claim 37, further comprising instructions that when executed cause the system to:

allow a first language module of the plurality of language modules to interact with a second language module of the plurality of language modules.

47. The machine readable medium of claim 37, wherein:

the first language module is operable to request the compilation of a specified portion of the computer program using the second language module.

48. The machine readable medium of claim 37, wherein:

at least one language module of the plurality of language modules is for Java language.

49. The machine readable medium of claim 37, further comprising instructions that when executed cause the system to:

extend a first language module of the plurality of language modules using a second language module of the plurality of language modules to provide a new language which is an extended version of a programming language compiled by the first language module.

50. The machine readable medium of claim 37, further comprising instructions that when executed cause the system to:

adopt tools to speed up the development of the plurality of language modules.

51. The machine readable medium of claim 37, further comprising instructions that when executed cause the system to:

utilize a compiler framework operable to perform a language-independent portion of the compilation process;

invoke a plurality of language modules, each language module in the plurality of language modules is operable to perform a language-dependent portion of a compilation process and to provide language information about the computer program;

provide an information interface operable to permit each client in a plurality of clients to interact with the compiler framework;

include a plurality of clients operable to utilize the information interface to request the compiler framework to perform at least one of the following tasks:

obtaining the language information produced by the plurality of language modules; and

requesting a service provided by the compiler framework.

52. The machine readable medium of claim 37, wherein the plurality of clients comprise:

an integrated development environment (IDE) for developing a computer program wherein the IDE uses the language information from the compiler framework to provide language features based on that information; and

a command-line interface for causing the compilation of the computer program and displaying any errors in the computer program.

53. The machine readable medium of claim 52, further comprising instructions that when executed cause the system to:

include a source code editor in the IDE to edit files in the computer program.

54. The machine readable medium of claim 51, further comprising instructions that when executed cause the system to:

allow a client to inform the compiler network of file changes; and

recompile the changed files and other files depend on them.

55. A system operable to perform a multi-programming-language compilation process on a computer program, comprising:

means to utilize a compiler framework operable to perform a programming language-independent portion of the compilation process;

means to invoke a plurality of language modules, wherein each language module is operable to perform a programming language-dependent portion of the compilation process; and

means to provide a plurality of language interfaces, wherein each language interface is operable to allow a language module to interact with the compiler framework.

56. A computer data signal embodied in a transmission medium, comprising:

a code segment including instructions to perform a multi-programming-language compilation process on a computer program, comprising:

utilizing a compiler framework operable to perform a programming language-independent portion of the compilation process;

invoking a plurality of language modules, wherein each language module is operable to perform a programming language-dependent portion of the compilation process; and

providing a plurality of language interfaces, wherein each language interface is operable to allow a language module to interact with the compiler framework.